HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P. O. Box 272400
Fort Collins, Colorado 80527-2400

PATENT APPLICATION

ATTORNEY DOCKET NO. __100111233-1__

IN THE
UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s): S. Brandon Keller

Application No.: 10/647,606

Filing Date: Aug. 25, 2003

Title: SYSTEM AND METHOD FOR DETERMINING CONNECTIVITY OF NETS IN A HIERARCHICAL CIRCUIT DESIGN

Confirmation No.: 2818

Examiner: Vuthe Siek

Group Art Unit: 2825

Mail Stop Appeal Brief-Patents
Commissioner For Patents
PO Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF

Sir:

Transmitted herewith is the Appeal Brief in this application with respect to the Notice of Appeal filed on __Dec. 15, 2005____.

The fee for filing this Appeal Brief is (37 CFR 1.17(c)) $500.00.

(complete (a) or (b) as applicable)

The proceedings herein are for a patent application and the provisions of 37 CFR 1.136(a) apply.

( ) (a) Applicant petitions for an extension of time under 37 CFR 1.136 (fees: 37 CFR 1.17(a)-(d) for the total number of months checked below:

( )	one month	$120.00
( )	two months	$450.00
( )	three months	$1020.00
( )	four months	$1590.00

( ) The extension fee has already been filled in this application.

(X) (b) Applicant believes that no extension of time is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

Please charge to Deposit Account **08-2025** the sum of ___**$500.00**___. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16 through 1.21 inclusive, and any other sections in Title 37 of the Code of Federal Regulations that may regulate fees. A duplicate copy of this sheet is enclosed.

(X) I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Alexandria, VA 22313-1450. Date of Deposit:__Feb. 15, 2006__
OR
( ) I hereby certify that this paper is being transmitted to the Patent and Trademark Office facsimile number_____ on _____

Number of pages: 22

Typed Name: Janet Ridpath

Signature:

Rev 12/04 (Aplbrief)

Respectfully submitted,

S. Brandon Keller

By _____

Curtis A. Vock

Attorney/Agent for Applicant(s)
Reg. No. **38,356**

Date: **Feb. 15, 2006**

Telephone No.: **(720) 931-3011**

IN THE UNITED STATES PATENT OFFICE

Applicant(s)	S. Brandon Keller, et al.	Examiner	Vuthe Siek
Serial No.	10/647,606	Group Art No.	2825
Filed	August 25, 2003	Confirmation No.	2818
For	System and Method for Determining Connectivity of Nets in a Hierarchical Circuit Design		

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

In accord with 37 CRF § 41.37, and fully responsive to the Office Action of September 19, 2005, Appellants hereby file their appeal brief in support of their Appeal in the above-identified matter (hereinafter the '606 Application). A notice of appeal, with appropriate fee of $500 as required by §§41.31, 41.20(b)(1), was filed on December 15, 2005. The $500 fee for this appeal brief, as required by 37 CRF §41.20(b)(2), is also filed herewith. This appeal brief is timely filed within two months of the mailing of the notice of appeal.

(1)   Real party in interest.

The real party in interest for this appeal is Hewlett-Packard Development Company, L.P. (HPDC), a limited partnership established under the laws of the State of Texas and having a principal place of business at 20555 S.H. 249, Houston, TX 77070, U.S.A. HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, L.L.C. Evidence of this assignment, which was recorded on October 6, 2003, may be found at reel/frame 014026/0117.

(2)   Related appeals and interferences.

No other appeals or interferences are currently known to Appellants that will directly affect, be directly affected by, or have a bearing on the decision to be rendered by the Board of Patent Appeals and Interferences in the present appeal.

(3)   Status of claims.

Claims 1-17 are pending in the '606 Application. Claims 1-17 stand rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Publication No.: 2003/0221173 (hereinafter "Fisher"). Applicants appeal all claims 1-17.

(4)   Status of amendments.

The '606 Application was filed on August 25, 2003. A first office action was mailed on April 6, 2005, to which a response was filed and entered July 6, 2005. On September 19, 2005, a final rejection was mailed, prompting the notice of appeal filed on December 15, 2005. Claims 1-17 are currently pending, of which claims 1-17 are original (without claim amendment during prosecution).

(5)   Summary of claimed subject matter.

In one embodiment (claim 1), a method determines connectivity of a hierarchical circuit design including: traversing hierarchical interface connections in a plurality of hierarchical blocks in at least a part of the circuit design (see for example paragraphs [0014-18] and step 210, Figure 2) by performing steps, for each block instance in each hierarchical block in the design (see for example paragraphs [0014-18] and step 220, Figure. 2), including, for each port instance on said each block instance (see for example paragraphs [0014-18] and step 230, Figure 2), wherein the port instance is not connected to a net in a parent block, generating a warning indicating the name of the port instance that is not connected (see for example paragraphs [0015-19] and step 250, Figure 2), and for each port, in each of the hierarchical blocks, that is not connected to a net within the block, generating a warning indicating the name of the port that is not connected (see for example paragraphs [0015-19] and step 250, Figure 2).

In another embodiment (claim 5), a method determines connectivity in a plurality of hierarchical blocks of a hierarchical circuit design, including traversing

hierarchical interface connections of the hierarchical blocks (see for example paragraphs [0014-19] and steps 210-250, Figure 2), wherein a top hierarchical level of one of the hierarchical blocks is selected as an initial hierarchical block(see for example paragraph [0014] and step 205, Figure 2).

In another embodiment (claim 7), a method determines connectivity of a hierarchical circuit design, including evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design (see for example paragraphs [0014-18] and step 210, Figure 2), whether each port instance, on said each block instance, is connected a net in a parent block (see for example paragraphs [0014], [0016-18] and step 232, Figure 2), and whether each port, in each of the hierarchical blocks, is connected to a net within the block (see for example paragraphs [0015-19] and step 240, Figure 2), such that a warning is generated upon detection of at least one disconnected said net within the hierarchical blocks (see for example paragraph [0019] and step 250, Figure 2).

In another embodiment (claim 10), a system determines connectivity of a hierarchical circuit design, including a processor, a connectivity module (see for example paragraph [0010] and connectivity module 111, Figure 1), executable by the processor (see for example paragraph [0010] and processor 102, Figure 1) to evaluate hierarchical interface connections between hierarchical blocks in the circuit design (see for example paragraphs [0010], [0014-19] and VLSI design 109, Figure 1, step 210, Figure 2), a user interface module (see paragraph [0010] and user interface module 112, Figure 1), coupled to the processor, for generating a warning upon detection of at least one disconnected net within the hierarchical blocks (see for example paragraphs [0015] and [0019] and step 250, Figure 2), wherein the connectivity module evaluates hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design (see for example paragraphs [0014-18] and step 220, Figure 2), whether each port instance (see for example paragraph [0013] and port instances 310, 312, 315, 317, ce2 and 320, Figure 3), on said each block instance, is connected a net in a parent block (see for example paragraphs [0014], and [0016-18] and steps 230, Figure 2), wherein a warning, indicating the name of each said port instance that is not connected, is

generated by the user interface module (see for example paragraph [0019] and step 250, Figure 2), and whether each port, in each of the hierarchical blocks, is connected to a net within the block (see for example paragraphs [0015-19] and step 240, Figure 2), wherein a warning, indicating the name of each said port that is not connected, is generated by the user interface module (see for example paragraph [0019] and step 250, Figure 2).

In another embodiment (claim 13), a system determines connectivity of a hierarchical circuit design, including means for evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design (see for example steps 210 and 220, Figure 2), whether each port instance, on said each block instance, is connected to a net in a parent block (see for example paragraphs [0014-18] and steps 220-232, Figure 2), and whether each port, in each of the hierarchical blocks, is connected to a net within the block (see for example paragraphs [0015-19] and step 240, Figure 2), and means for generating a warning upon detection of at least one disconnected said net within the hierarchical blocks (see for example paragraphs [0015-19] and step 250, Figure 2).

In another embodiment (claim 17) a software product comprises instructions, stored on computer-readable media (see for example paragraph [0020]), wherein the instructions, when executed by a computer, perform steps for determining connectivity of a hierarchical circuit design, including: evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design, whether each port instance, on said each block instance, is connected to a net in a parent block (see for example paragraphs [0014], and [0016-18] and step 220, Figure 2), and whether each port, in each of the hierarchical blocks, is connected to a net within the block (see for example paragraphs [0015], and [0016-18] and steps 240, Figure 2), and generating a warning upon detection of at least one disconnected said net within the hierarchical blocks (see for example paragraphs [0014-15] and [0018-19] and step 250, Figure 2).

(6)   **Grounds for rejection to be reviewed on appeal.**

Whether claims 1-17 are rendered obvious by Fisher in accordance 35 U.S.C. §103(a).

(7)   Arguments.

Illustratively, the '606 Application at least teaches a method for determining connectivity of a hierarchical circuit design by evaluating connectivity of each port and port instance of each block instance within the circuit design to a parent block net and a net within the block instance. If any port or port instance is not connected to a net, one or more warning messages are generated. The '606 Application does not traverse nets.

On the other hand, Fisher discloses analyzing connectivity conditions in a netlist data file; but Fisher does not traverse the circuit design based upon hierarchical blocks, block instances, ports and port instances. Specifically, Fisher "loops through all nets associated with each non-leaf cells*[sic]* and builds a list of all leaf cells connected to each net." See Fisher paragraph [0023] and steps 57 and 58, FIG. 3.

As noted in paragraph [0002] of the '606 Application, "tracing each net in a design... is undesirably slow." The '606 Application does not traverse nets of the circuit design to identify unconnected ports and port instances.

Clearly, the method of Fisher is different from that of the '606 Application. For each non-leaf cell, Fisher creates a collection of nets. Then, for each of these nets, Fisher creates a list of all leaf-cells connected to the net. Specifically, Fisher "determines the number of leaf cell inputs and leaf cell outputs connected to a given net, and from that, determines whether certain defects exist in the net list file." See Fisher paragraph [0026].

Fisher discloses that a zero-connect port is a port of a given block that connects to no leaf cells. However, Fisher determines this 'port' connectivity based upon connectivity of leaf cells to a net. Thus, in Fisher, a zero-connect port may connect to a net, but the net does not connect to a leaf cell. Therefore, Fisher does not specifically identify ports that are not connected to a net.

Paragraph [0012] of the '606 Application teaches that "a 'port/portinst' thus comprises two contiguous parts, a first part, termed a 'portinst', which is a port instance located externally on a box (or instance) boundary; and a second part, termed a 'port', which is located internally on the box (or block) boundary." See also

paragraph [0013] of the '606 Application. Clearly, this distinction between a port and a portinst allows the '606 Application to provide additional information regarding connectivity issues within its warning messages. Fisher does not distinguish between ports and port instances and therefore cannot evaluate and generate warning messages for both ports and port instances.

For at least these reasons, Fisher cannot render obvious any of independent claims 1, 5, 7, 10, 13 and 17, since Fisher does not teach or suggest each claim element of these claims.

Further, it would not have been obvious to modify Fisher to traverse block instances, port and port instances in place of nets since Fisher operates to identify connectivity issues such as gate-only net, potential drive-fight, floating nets and port direction mismatch and thus requires net traversal and such modification would render Fisher inoperable. Assuming, arguendo, Fisher is modified to iterate through hierarchical blocks, block instances, ports and port instances, there would be insufficient information pertaining to each port and port instance to identify connectivity issues such as gate-only net, potential drive-fight, floating nets. Therefore, there would be no motivation to modify Fisher.

More particularly, claim 1 recites a method for determining connectivity of a hierarchical circuit design, including steps of:

a) traversing hierarchical interface connections in a plurality of hierarchical blocks in at least a part of the circuit design by performing steps, for each block instance in each hierarchical block in the design, including:

    i. for each port instance on said each block instance, wherein the port instance is not connected to a net in a parent block, generating a warning indicating the name of the port instance that is not connected; and

    ii. for each port, in each of the hierarchical blocks, that is not connected to a net within the block, generating a warning indicating the name of the port that is not connected.

Step a) of claim 1 thus requires that hierarchical interface connections are traversed, and not nets. Further sub-step i generates a warning indicating the name of a port instance that is not connected and sub-step ii generates a warning indicating the name of a port that is not connected. Ports and port instances are different entities that comprise hierarchical interface connections. See for example, paragraphs [0012-13] of the '606 Application.

On the other hand, Fisher reaches its result by processing nets and not hierarchical interface connections. Further, Fisher does not disclose port instances. Specifically, Fisher makes no distinction between connections within a block and connections external to a block.

As noted above and required by sub-steps i and ii, the '606 Application operates based on ports and port instances to identify ports and port instances that do not connect to nets. On the other hand, Fisher operates upon nets and only evaluates connectivity of these nets to leaf cells. Fisher generates a connectivity report but makes no disclosure as to the contents of the report. Fisher makes no disclosure of indicating non-connectivity of a specific port or port instance. In fact, since Fisher operates based upon connectivity of nets to leaf cells, Fisher cannot identify ports or port instances that are not connected to a net; Fisher only processes nets of non-leaf cells. See Fisher items 56-59 of FIG. 2 and associated description. Further, teaching away from the '606 Application, the method of Fisher identifies a net that does not connect to any leaf cells as a zero connect port. The '606 Application does not generate warning for ports or port instances that connect to a net.

Claims 2-4 depend from claim 1 and benefit from like argument.

In addition, claim 4 recites that a top hierarchical level of one of the hierarchical blocks is selected as an initial hierarchical block in the traversing step. Teaching away from claim 4, Fisher discloses that "during execution, the defect detection algorithm 20 reads the hierarchical BDL netlist file from the database 25 beginning with the top level parent BDL block." See Fisher paragraph [0019]. Fisher does not disclose selecting a hierarchical block as an initial hierarchical block.

Claim 5 recites a method for determining connectivity in a plurality of hierarchical blocks of a hierarchical circuit design, including steps of:

a) traversing hierarchical interface connections of the hierarchical blocks, wherein a top hierarchical level of one of the hierarchical blocks is selected as an initial hierarchical block, by performing steps, for each block instance in each hierarchical block in at least a part of the design, including:

    i. for each port instance on said each block instance, wherein the port instance is not connected a net in a parent block, generating a warning indicating the name of the port instance that is not connected; and

    ii. for each port, in each of the hierarchical blocks, that is not connected to a net within the block, generating a warning indicating the name of the port that is not connected;

b) wherein the warning comprises a message transmitted to a user terminal.

Text within claim 5 is similar to the text used in claim 1 and therefore benefits from the same arguments presented above.

Claim 6 depends from claim 5 and benefits from like argument.

Claim 7 recites a method for determining connectivity of a hierarchical circuit design, including steps of:

a) evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design:

    i. whether each port instance, on said each block instance, is connected a net in a parent block; and

    ii. whether each port, in each of the hierarchical blocks, is connected to a net within the block; and

b) generating a warning upon detection of at least one disconnected said net within the hierarchical blocks.

Text within claim 7 is similar to the text used in claim 1 and therefore benefits from the same arguments presented above.

Claim 8 depends from claim 7 and benefits from like argument.

Claim 9 (depending from claim 7) recites that a top hierarchical level of one of the hierarchical blocks is selected as an initial hierarchical block in the evaluating step. Claim 9 is similar to claim 4 and benefits from like argument.

Claim 10 recites a system for determining connectivity of a hierarchical circuit design, including:

a) a processor;

b) a connectivity module, executable by the processor to evaluate hierarchical interface connections between hierarchical blocks in the circuit design;

c) a user interface module, coupled to the processor, for generating a warning upon detection of at least one disconnected net within the hierarchical blocks;

d) wherein the connectivity module evaluates hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design:

    i. whether each port instance, on said each block instance, is connected a net in a parent block, wherein a warning, indicating the name of each said port instance that is not connected, is generated by the user interface module; and

    ii. whether each port, in each of the hierarchical blocks, is connected to a net within the block, wherein a warning, indicating the name of each said port that is not connected, is generated by the user interface module.

Element b) of claim 10 thus requires that a connectivity module traverses hierarchical interface connections, and not nets. Further sub-step i of element d) generates a warning indicating the name of a port instance that is not connected and sub-step ii of element d) generates a warning indicating the name of a port that is not

connected. Ports and port instances are different entities that comprise hierarchical interface connections. See for example, paragraphs [0012-13] of the '606 Application.

On the other hand, Fisher reaches its result by processing nets and not hierarchical interface connections. Further, Fisher does not disclose port instances. Specifically, Fisher makes no distinction between connections within a block and connections external to a block.

As noted above and required by sub-steps i and ii, the '606 Application operates based on ports and port instances to identify ports and port instances that do not connect to nets. On the other hand, Fisher operates upon nets and only evaluates connectivity of these nets to leaf cells. Fisher generates a connectivity report but makes no disclosure as to the contents of the report. Fisher makes no disclosure of indicating non-connectivity of a specific port or port instance. In fact, since Fisher operates based upon connectivity of nets to leaf cells, Fisher cannot identify ports or port instances that are not connected to a net; Fisher only processes nets of non-leaf cells. See Fisher items 56-59 of FIG. 2 and associated description. Further, teaching away from the '606 Application, Fisher identifies a net that does not connect to any leaf cells as a zero connect port. Claim 10 is not concerned with generating a warning for ports or port instances that connect to a net.

Claims 11 and 12 depend from claim 10 and benefits from like argument.

In addition, claim 12 recites that a hierarchical model of the circuit design, accessible by the connectivity module via the processor, is used to indicate the hierarchical interface connections between hierarchical blocks in the circuit design. Claim 12 requires that a hierarchical model be used to indicate the hierarchical interface connections between hierarchical blocks in the circuit design. Fisher on the other hand makes no such disclosure. In fact, teaching away from claim 12, Fisher discloses that "a tool is provided that traverses the netlist file prior to a simulation model and design layout, or artwork, being created". See Fisher paragraph [0016].

Claim 13 recites a system for determining connectivity of a hierarchical circuit design, including:

    a)   means for evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design:

        i.   whether each port instance, on said each block instance, is connected to a net in a parent block; and

        ii.   whether each port, in each of the hierarchical blocks, is connected to a net within the block; and

        iii.   means for generating a warning upon detection of at least one disconnected said net within the hierarchical blocks.

Element a) of claim 13 thus requires that hierarchical interface connections are traversed, and not nets. Specifically, sub-step i determines whether port instances are connected, sub-step ii determines whether ports are connected and sub-step iii generates a warning for disconnected nets. Ports and port instances are different entities that comprise hierarchical interface connections. See for example, paragraphs [0012-13] of the '606 Application.

On the other hand, Fisher reaches its result by processing nets and not hierarchical interface connections. Further, Fisher does not disclose port instances. Specifically, Fisher makes no distinction between connections within a block and connections external to a block.

As noted above and required by sub-steps i and ii, the '606 Application operates based on ports and port instances to identify ports and port instances that do not connect to nets. On the other hand, Fisher operates upon nets and only evaluates connectivity of these nets to leaf cells. Fisher generates a connectivity report but makes no disclosure as to the contents of the report. Fisher makes no disclosure of indicating non-connectivity of a specific port or port instance. In fact, since Fisher operates based upon connectivity of nets to leaf cells, Fisher cannot identify ports or port instances that are not connected to a net; Fisher only processes nets of non-leaf cells. See Fisher items 56-59 of FIG. 2 and associated description. Further, teaching away from the '606 Application, Fisher identifies a net that does not connect to any

leaf cells as a zero connect port. Claim 13 is not concerned with generating warnings for ports or port instances that connect to a net.

Claims 14-16 depend from claim 13 and benefit from like argument.

In addition, claim 14 recites that a top hierarchical level one of the hierarchical blocks is selected as an initial hierarchical block used by said evaluating means. Teaching away from claim 14, Fisher discloses that "during execution, the defect detection algorithm 20 reads the hierarchical BDL netlist file from the database 25 beginning with the top level parent BDL block." See Fisher paragraph [0019]. Fisher does not disclose selecting a hierarchical block as an initial hierarchical block.

Claim 17 recites a software product with instructions, stored on computer-readable media, wherein the instructions, when executed by a computer, perform steps for determining connectivity of a hierarchical circuit design, including:

a) evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design:

    i. whether each port instance, on said each block instance, is connected to a net in a parent block; and

    ii. whether each port, in each of the hierarchical blocks, is connected to a net within the block; and

    iii. generating a warning upon detection of at least one disconnected said net within the hierarchical blocks.

Element a) of claim 17 thus requires that hierarchical interface connections are traversed, and not nets. Specifically, sub-step i determines whether port instances are connected, sub-step ii determines whether ports are connected and sub-step iii generates a warning for disconnected nets. Ports and port instances are different entities that comprise hierarchical interface connections. See for example, paragraphs [0012-13] of the '606 Application.

On the other hand, Fisher reaches its result by processing nets and not hierarchical interface connections. Further, Fisher does not disclose port instances.

Specifically, Fisher makes no distinction between connections within a block and connections external to a block.

As noted above and required by sub-steps i and ii, the '606 Application operates based on ports and port instances to identify ports and port instances that do not connect to nets. On the other hand, Fisher operates upon nets and only evaluates connectivity of these nets to leaf cells. Fisher generates a connectivity report but makes no disclosure as to the contents of the report. Fisher makes no disclosure of indicating non-connectivity of a specific port or port instance. In fact, since Fisher operates based upon connectivity of nets to leaf cells, Fisher cannot identify ports or port instances that are not connected to a net; Fisher only processes nets of non-leaf cells. See Fisher items 56-59 of FIG. 2 and associated description. Further, teaching away from the '606 Application, Fisher identifies a net that does not connect to any leaf cells as a zero connect port. Claim 17 is not concerned with generating warnings for ports or port instances that connect to a net.

In view of the above arguments, Fisher is clearly different from claims of the '606 Application. Further, in view of the operational requirement of Fisher to prove nets, it would not be obvious to modify Fisher to iterate through hierarchical blocks, block instances, ports and port instances since Fisher cannot operate without traversing nets.

(8)   Claims Appendix.

Appellants enclose a copy of the claims involved in this appeal as an appendix hereto.

(9)   Evidence Appendix.

No evidence is entered or relied upon in this appeal.
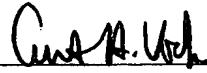
(10)   Related Proceedings Appendix.

To Appellants' knowledge, there are no decisions rendered by a court or the Board for submission with this appeal.

Conclusions

Appellants respectfully submit that the claims 1-17 are not rendered obvious by the art of record. Other than the costs for the appeal brief, we believe no additional fees are due in connection with this matter. However, if any additional fee is deemed necessary, the Commissioner is hereby authorized to charge such fee to Deposit Account No. 08-2025.

Respectfully submitted,

By: _____

Curtis A. Vock, Reg. No. 38,356
LATHROP & GAGE L.C.
4845 Pearl East Circle, Suite 300
Boulder, CO 80301
Telephone: (720) 931-3011
Facsimile: (720) 931-3001

CLAIM APPENDIX TO APPEAL BRIEF

1.   A method for determining connectivity of a hierarchical circuit design, comprising steps of:

traversing hierarchical interface connections in a plurality of hierarchical
blocks in at least a part of the circuit design by performing steps, for
each block instance in each hierarchical block in the design, including:
for each port instance on said each block instance, wherein the
port instance is not connected to a net in a parent block,
generating a warning indicating the name of the port
instance that is not connected; and
for each port, in each of the hierarchical blocks, that is not
connected to a net within the block, generating a warning
indicating the name of the port that is not connected.

2.   The method of claim 1, wherein the traversing step is performed prior to an analysis of the circuit design.

3.   The method of claim 1, wherein the warning comprises a message transmitted to a user terminal.

4.   The method of claim 1, wherein a top hierarchical level of one of the hierarchical blocks is selected as an initial hierarchical block in the traversing step.

5.   A method for determining connectivity in a plurality of hierarchical blocks of a hierarchical circuit design, comprising steps of:

traversing hierarchical interface connections of the hierarchical blocks,
wherein a top hierarchical level of one of the hierarchical blocks is
selected as an initial hierarchical block, by performing steps, for each
block instance in each hierarchical block in at least a part of the design,
including:

for each port instance on said each block instance, wherein the port instance is not connected a net in a parent block, generating a warning indicating the name of the port instance that is not connected; and

for each port, in each of the hierarchical blocks, that is not connected to a net within the block, generating a warning indicating the name of the port that is not connected;

wherein the warning comprises a message transmitted to a user terminal.

6. The method of claim 5, wherein the traversing step is performed prior to an analysis of the circuit design.

7. A method for determining connectivity of a hierarchical circuit design, comprising steps of:

evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design:

whether each port instance, on said each block instance, is connected a net in a parent block; and

whether each port, in each of the hierarchical blocks, is connected to a net within the block; and

generating a warning upon detection of at least one disconnected said net within the hierarchical blocks.

8. The method of claim 7, wherein the warning comprises a message transmitted to a user terminal.

9. The method of claim 7, wherein a top hierarchical level of one of the hierarchical blocks is selected as an initial hierarchical block in the evaluating step.

10. A system for determining connectivity of a hierarchical circuit design, comprising:

a processor;

a connectivity module, executable by the processor to evaluate hierarchical interface connections between hierarchical blocks in the circuit design;

a user interface module, coupled to the processor, for generating a warning upon detection of at least one disconnected net within the hierarchical blocks;

wherein the connectivity module evaluates hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design:

whether each port instance, on said each block instance, is connected a net in a parent block, wherein a warning, indicating the name of each said port instance that is not connected, is generated by the user interface module; and

whether each port, in each of the hierarchical blocks, is connected to a net within the block, wherein a warning, indicating the name of each said port that is not connected, is generated by the user interface module.

11.   The system of claim 10, further including a storage unit, accessible to the processor, in which the circuit design is stored is stored.

12.   The system of claim 10, wherein a hierarchical model of the circuit design, accessible by the connectivity module via the processor, is used to indicate the hierarchical interface connections between hierarchical blocks in the circuit design.

13.   A system for determining connectivity of a hierarchical circuit design, comprising:

means for evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design:

whether each port instance, on said each block instance, is connected to a net in a parent block; and

whether each port, in each of the hierarchical blocks, is connected to a net within the block; and

means for generating a warning upon detection of at least one disconnected said net within the hierarchical blocks.

14.   The system of claim 13, wherein a top hierarchical level one of the hierarchical blocks is selected as an initial hierarchical block used by said evaluating means.

15.   The system of claim 13, wherein the warning comprises a message transmitted to a user terminal.

16.   The system of claim 13, wherein the hierarchical interface connections are evaluated prior to an analysis of the circuit design.

17.   A software product comprising instructions, stored on computer-readable media, wherein the instructions, when executed by a computer, perform steps for determining connectivity of a hierarchical circuit design, comprising:

evaluating hierarchical interface connections of the design by determining, for each block instance in each of the hierarchical blocks in the design: whether each port instance, on said each block instance, is connected to a net in a parent block; and whether each port, in each of the hierarchical blocks, is connected to a net within the block; and

generating a warning upon detection of at least one disconnected said net within the hierarchical blocks.

Evidence Appendix

(BLANK)

Related Proceedings Appendix

(BLANK)